Over-Constraint QoS Routing in Large Networks

W. S. Goodridge† W. R. Robertson†W. P. Phillips† and Shyamala Sivakumar‡

† Dalhousie University, ‡ St. Mary's University

Abstract: Routing using more than one metric is difficult. Distributed Multimedia applications have stringent QoS demands on the network. However, many current QoS routing algorithms have large execution times when strict (small) user constraints are used. We present an algorithm that have relatively small execution times under strict constraints and demonstrate via simulation that this is so.

1 Introduction.

Real-time network applications like telemedicine, video conferencing, and virtual reality will demand highest quality of service (QoS) and reliability from the data transport infrastructure. The networking community has developed frameworks such as InServ and Diffserv to alter performance for network flows by changing the schedule in a router or switch. However, there is another way to achieve QoS, which is called QoS Routing and the goal is to select a feasible path for a network flow based on the requirements of the flow. A lot of research has been done on QoS Routing and three main challenges are evident: firstly, finding a feasible path is a very complex problem; secondly, the routing protocols will have to measure and exchange values for multiple network metrics such as delay, jitter and bandwidth; and thirdly, a mechanism for the user or application to communicate a set of user constraints to the network is needed. In this paper we will only consider the challenge of finding feasible paths in communication networks.

Finding a path subject to multiple metrics is difficult and is considered NP-complete [1]. Since the motivation for QoS Routing is rooted in the fact that multimedia applications require stringent user constraints, a good question is whether or not existing QoS routing algorithms can address the needs of applications that have strict constraints (over- constraints [2]). In this paper we will give a background on some of the QoS routing algorithms in section 2, and in Section 3 answer this important question. In Section 4, we introduce an alternative QoS Routing algorithm specially designed to work in over-constraint environments. In Section 5, we present simulations to support the claim that our algorithm works well in large networks.

2. Existing QoS Routing Algorithms

Before we get into existing QoS algorithms we will give a formal definition for two problems associated with QoS Routing. Consider a graph $G(N;E)$ consisting of a set of additive metrics $w_i(e)$ for each link $e \in E$, and user requested constraints L_i, $i \in [1, m]$. The goal of multi-path-constrained routing (MCP) is to find a path P from source node s to destination node t such that $w_i(P) \leq L_i$ for all i. A path that satisfies all m constraints is often referred to as a feasible path. There may be multiple paths in the graph $G(N; E)$ that satisfy the constraints, any of which are solutions to the MCP problem. Hence it may be necessary to use some type of optimization criteria to select a path from the set of feasible paths. This more difficult problem is called the multi-constrained optimal path (MCOP) problem.

Because both the MCP and MCOP problems are NP-complete the bulk of the solutions proposed are heuristics. These heuristics algorithms include Jaffe [3], Chen [4], Iwata [5], and TAMCRA [6]. Typically these algorithms are highly specialized and cannot adapt easily to a wide range of user requirements.

There also exist exact solutions for the MCP/MCOP like the SAMCRA [6] and H_MCOP [7]. These algorithms offer good performance at the expense of possible high complexities and running times growing exponentially in the worst case. However, in [8] it is argued that in practice an exact algorithm may work in polynomial time, making guaranteed QoS possible.

Most of the heuristic solutions involve the use of a modified shortest path algorithm (SPA). However, shortest path algorithms use a single metric and therefore these solutions tend to be complicated in their effort to accommodate multiple metrics. On the other hand, most exact algorithms use the k shortest paths with some objective function to select the best path.

3. Over-Constraint users requests

From [8] it can be inferred that the NP-complete behaviour of a given network depends on its link topology, link weight structure and user constraints. The choice of user constraints can heavily influence how many feasible paths exist and therefore will affect the execution time needed to find an optimal path. User constraints can be strict (over-constraint) [2] or loose (under-constraint). The set of strict user constraints [9] can be defined by equation 1.

$$L_i = w_i(P) + \phi,\ i = 1,\ldots,m \qquad (1)$$

In equation 1, P is the path for which $max(w_i(P^*))$ is minimum where P^* is the set of all paths connecting s and t. Similarly, a set of loose constraints can be defined by equation 1 where P is the path for which $max(wi(P^*))$ is maximum. Results of simulations done in [9] suggested that most heuristic and exact algorithms provide near-optimal success rates [9] with small execution times when loose user constraints are used. However, under strict user constraints using the Waxman graph class, heuristic algorithms preformed poorly with respect to their success ratios. However, for this class of graph execution times increased but remained polynomial with respect to the network size for both heuristic and exact algorithms. For the Lattice class of graphs exact algorithms experienced exponential execution time growth rates with respect to the network size.

The motivation for QoS Routing is rooted in the fact that multimedia applications require stringent user constraints. Based on the simulations results in [9] we think that existing algorithms are inefficient under relatively strict user constraints.

4. An algorithm for over-constraint QoS-Routing

We present an exact algorithm called Routing Decision Support System (RDSS) that does not depend on a shortest path algorithm. There are two main components to this algorithm:

1. Using a modified version of the all-path algorithm called Constraint All-path (CAP) algorithm to find all paths meeting the user constraints. Figure 1 shows the pseudocode for the CAP algorithm. The algorithm accepts a graph *(G)*, source*(s)*, destination *(t)*, a set of user constraints *(C)* and returns a list (*allpaths*) of all the paths between s and t. The data structure for the node consists of a path from the source to the node, the height of the node in the tree and a node identifier called vertex.

   The main advantage in this approach is that under strict user constraints the queue size of *kN* can be significantly reduced by eliminating sub-trees violating the user constraints.

2. Using the RDSS to find a path (from the set of paths resulting from step 1) that most closely match the user or optimization requirements. The RDSS algorithm does this by taking the constraint paths produced from the CAP algorithm and building preference functions [10] for each metric. In [11] we presented introductions to this approach. The basic idea is that we use a preference function that accepts a network metric value say *x* as a parameter and returns a value, *s(x)* between –1 and 1 that represents the preference value of *x* relative to all the network values for this metric.

5. Simulation

Our simulation will focus on the computational aspect of the RDSS algorithm under a wide range of strict user constraints. To do this we use Waxman graphs [9], with negatively correlated link weights representing delay and cost. For 10 different networks containing 400 nodes each, we executed the Jaffe, SAMCRA and RDSS algorithms to find a feasible path between node 0 and node 399. Strict constraints for delay and cost values each ranging from 100 to 150 are used. Since the same delay and

cost value for a given time measurement are used we represent results using a 2-dimensional graph. Figure 1, shows the results for one of the networks tested, the same pattern is observed for other networks From the graph, it is evident that the RDSS algorithm has relatively small execution times for delay/cost values between 103 and 135. However, between delay/cost 135 and 152 the Jaffe and RDSS algorithms execution times are closer. The Jaffe algorithm although having an average execution time of 260 ms did not produce feasible paths for the user constraints.

```
findAllPath(G, s, t, allpaths, C)     {
  node ←new Node(path ← Null, height ←0)
  queue.enqueue(node)
  while (not queue.empty) {
   closest ← queue.dequeue
    if (closest.vertex == t)
      allpaths.append(closest.Path)
       Relax (closest)
}
```

```
Relax (closest) {
  HEIGHT <- closest.height + 1
    for each neightbor w of closest
       PATH ← node.path + w
         if (PATH.w_i ≤ C_i for all i) {
           node ← new Node(path ← PATH, height ← HEIGHT)
           queue.enqueue(node)
         }
   }
}
```

Fig. 1 The Constraint All-Paths algorithm that is used to generate constraint paths

1. For a given metric say m_j select all metric values m_{ij} for routes R_i, where $i = \{1,,2,...,k\}$

2. For a given metric m_j the best value from the set of all routes for m_j is assigned to a_{ij} where i is the route corresponding to the best value for metric m_j.

3. For a given metric m_j the worse value from the set of all routes for m_j is assigned to b_{ij} where i is the route corresponding to the worse value for metric m_j.

4. A preference scale is then derived for each metric with the following properties: a. If $x \in [b, a]$ then a scale is define using equation 2.

$$sj(x) = \begin{cases} \left(2\dfrac{(x-b)}{(a-b)} - 1\right) & a \neq b \\ 1 & a = b \end{cases} \qquad (2)$$

b. If m_j is concave and $x < b$ then $s(x) = -1$ and if and $x > a$ then $s(x)=1$. Additionally, for additive and multiplicative metrics if $x < b$ then $s(x)=0$ and if $x > a$ and $s(x) = 1$.

5. Let A be a k x m matrix (m = metrics, k = paths) containing columns $s_1(x_1)$, $s_2(x_2)$, …, $s_m(x_m)$ where $s_j(x_j)$ represents a scale for each metric m_j.

6. To find the closest match to the given user constraints the following conversion is done: $v = [s_1(c_1) \quad s_2(c_2) \quad \ldots \quad s_k(c_k)]$.

7. The $Av = \overline{o}$ matrix multiplication is then performed which results in a k x 1 vector. Vector $\overline{o}$ is examined for the highest value of the vector at position y where $1 \leq y \leq k$. The value of y corresponds to the route that will be a feasible path.

8. To find an optimal path for a constraint C_i the $s(c_i)$ is set to 1 and step 7 is performed.

Fig 2. RDSS algorithm for finding an optimal path given a set of constraint paths

7. Conclusion

The RDSS algorithm seems to be relatively fast for networks when strict user constraints are specified. This means that the algorithm can be used to find feasible routes for packets of stringent multimedia applications. However, the presented RDSS algorithm has exponential times under loose constraints. We will address this problem in a future paper.

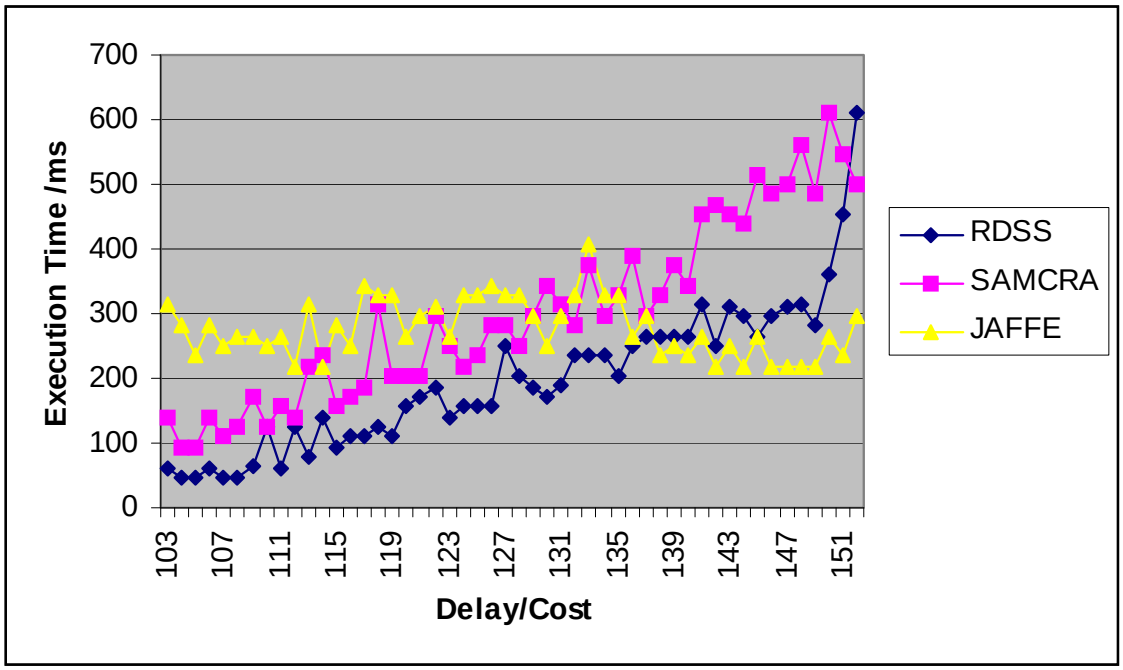

Fig. 3 2D graph with delay and cost having the same values on the x-axis and execution times on y-axis for RDSS, SAMCRA and Jaffe algorithms.

References.

[1] Z. Wang and J. Crowcroft, "Quality-of-Service Routing for Supporting Multimedia Applications", IEEE Journal on Selected Areas in Communications, vol. 14, n. 7, pp. 1228-1234, 1996

[2] Van Mieghem, P. and F. A. Kuipers, 2003, The Impact of Correlated Link Weights on QoS Routing, Computer Communications, vol. 26, No. 4, March 2003, pp. 376-387.

[3] Jaffe, J.M. , "*Algorithms for Finding Paths with Multiple Constraints*", Networks, vol.14, pp.95-116, 1984.

[4] Chen, S. and K. Nahrstedt, 1998a, "*On Finding Multi-constrained Paths*", Proc. ICC '98, Atlanta, Georgia.

[5] Iwata, A., R. Izmailov, D.-S. Lee, B. Sengupta, G. Ramamurthy and H. Suzuki, 1996, "*ATM Routing Algorithms with Multiple QoS Requirements for Multimedia Internetworking*", IEICE Trans. Commun., vol. E79-B, no. 8, August.

[6] P. Van Mieghem, H. De Neve and F.A. Kuipers, "Hop-by-Hop Quality of Service Routing", Computer Networks, Vol. 37, No. 3-4, pp. 407–423, November 2001.

[7] T. Korkmaz, M. Krunz, "Multi-Constrained Optimal Path Selection", in Proceedings of the IEEE INFOCOM 2001 Conference, Vol. 2, pp. 834–843. Anchorage, Alaska, April 2001.

[8] Van Mieghem, P. and F. A. Kuipers, 2003, On the Complexity of QoS Routing, Computer Communications, vol. 26, No. 4, March 2003, pp. 376-387.

[9] Kuipers, F. A., T. Korkmaz, M. Krunz and P. Van Mieghem, 2004, Performance Evaluation of Constraint-Based Path Selection Algorithms, to appear in IEEE Network.

[10] J. Barzilai, "Notes on measurement and Decision Theory," Proceedings of the NSF Design and Manufacturing Research Conference, San Juan, Puerto Rico, pp.1-11, 2002

[11] W. Goodridge, W. R. Robertson, W. P. Phillips and S, Sivakumar , "Comparing a Novel QoS Routing Algorithm to Standard Pruning Techniques used in QoS Routing Algorithms", IEEE CCECE 2004, Niagara Falls, May 2004